DIAGNOSTIC PROGRAM
FOR THE
XEBEC 1223-A SERIES
FLEXIBLE DISK CONTROLLER

[illegible]

October 22, 1976

Noted
No more
Test 21

TABLE OF CONTENTS

7. DIAGNOSTIC PROGRAM

7.1 General

This diagnostic test program confirms proper output, input, and control functions for the PDP-8/XEBEC 1200/ Floppy Disk System. The test operator runs the program under the default mode or defines his own tests. Up to six drives can be checked serially. This diagnostic does not check more than one controller, nor more than one CPU interface.

7.2 Hardware Requirements

This diagnostic program requires a PDP-8 computer with at least 4096 words of memory and a console teletype. The upper 200_8 words of memory in field 0 are reserved for loaders and are not overwritten.

7.3 Input Conventions

This diagnostic program performs all the communication between the operator and the program via the console teletype and, in some cases, the console switch register. All the replies to specific input requests are entered from the keyboard and are terminated by the RETURN key. This program will operate using the 64 printing characters, RUBOUT, RETURN, and the 'break' characters: Control-F, Control-P, Control-E, Control-T and Control-N. Two characters are designated for line editing:

RUBOUT	Deletes one character to the left for each input and echoes as a back slash '\'

@	Deletes the entire line to the left and echoes as '@'

The break characters are used to alter the execution sequence of the diagnostic as follows:

Control-F	Echoes as '↑F' and returns to the formatter prompt, 'FMPR='.
Control-P	Echoes as '↑P' and returns to the pack prompt, 'PACK='.
Control-E	Echoes as '↑E' and returns to the error mask prompt, 'EMSK='.
Control-T	Echoes as '↑T' and returns to the test prompt, 'TEST='.
Control-N	Echoes as '↑N', stops execution of the test in progress, and starts execution of the next test in the test sequence.

The diagnostic prompts form a sequence. The break characters may be used to request a reprompt as far back in the sequence as necessary.

All numeric values input or output are in octal. If more than 4 digits are input for a numeric value the program will recognize only the low order 12 binary bits (4 octal digits) of the value specified.

If any operator reply to an input request cannot be recognized by the program or is outside the allowed set of values, the program will output the error message '??' and repeat the request. Similarly, if more than the allowable number of parameters are entered, the error message '??' will be displayed and the request repeated.

7.4 Operating Procedure

7.4.1 Preparation

1. The console teletype must be in the LINE position. Each disk drive to be tested should be loaded with a diskette.
2. Load the program into the PDP-8 memory.

7.4.2 Running the Program:

1. Start the program at memory location 200_8. The program will respond with a four line output describing the location and size of the I/O buffers, as well as the number of fields of memory.

   Example:

   ```
   WBUF = 6360
   RBUF = 6770
   BUFL =  410
   LAST MEM. FIELD = 3
   ```

   indicates that the write buffer (WBUF) starts at 6360_8, the read buffer (RBUF) starts at 6770_8, the lengths of the buffers (BUFL) are 410_8, and the last memory field (LAST MEM. FIELD) is 3.

2. The program requests operational parameters from the test operator. As each set of parameters is requested the operator will enter the data via the console teletype. Each request will have a unique prompt. The prompts, expected replies, and optional replies are given below.

The optional parameters are enclosed within less than (<) and greater than (>) symbols. The prompt is underlined in the following discussion for clarity but is not underlined when the diagnostic is run. The RETURN key terminates all the messages.

7.4.3 Prompt Sequence:

7.4.3.1 FMPR= <device address>

The program will request the device address of the controller. The address must be in the range 1 to 76_8 and defaults to 30_8.

7.4.3.2 PACK=DRIVE/TRACK(S)

The program will request the identity and sequence of drives to be tested. Up to 8 definitions may be entered and are delimited by commas (,). DRIVE denotes the drive number and must be in the range 0 to 5. TRACK(S) denotes the track or track boundaries. Boundaries are entered with the lower boundary first and separated by a hyphen (-). They must satisfy the relation $0 \leq \text{lower} \leq \text{upper} \leq 114_8$. Different track ranges on the same drive may be specified.

Examples: PACK=3/0-10 denotes drive 3, tracks 0 to 10_8
PACK=1/76, 3/0-10 denotes drive 1, track 76_8, and drive 3, tracks 0 to 10_8.
PACK=3/0-10, 3/70-76 denotes tracks 0 to 10_8 and 70_8 to 76_8 both ranges on drive 3.
PACK=U Λ'U' as a response to this prompt will enter the User Specified Test mode.

7.4.3.3 EMSK=<mask> ,<detail line limit>

The program will request the parameters for error message control. Mask is the error status mask. The operator may inhibit the reporting of a status error by placing a 0 in the appropriate bit position. For example: EMSK=7677 will inhibit reporting of Timeout (TMO) error massages. The mask defaults to 7777_8. Detail line limit specifies the maximum number of lines to report when a data error occurs. A limit of 0 will inhibit all verify error message printout. The default of 10_8 will limit the printing to 8 lines for each sector found in error.

7.4.3.4 TEST=TEST SEQUENCE (S)

The program will conclude its prompting for parameters by requesting the test sequence to be performed. The operator can enter up to 72 characters of test information. The information allows specification of the order and number of repetitions of tests for groups of tests. Test sequences are delimited by commas(,). Each test group (G) may be repeated a specified number of times (C). The general form is: <C*>G where G may be defaulted to the test sequence 1 through 20_8, or may specify one particular test or a test range. A test range is inclusive and separated by a hyphen, e.g. 3-7 denotes tests 3, 4, 5, 6, and 7. C is used to specify the number of repetitions of G. If C is 0 the sequence is constantly repeated. Test groups or test sequences may be nested in parentheses. Thus, 2*(1-3,5-15), indicates that the sequences 1 to 3 and 5 to 15_8 will be performed twice. 2*(2*1-3,5-15) indicates that the sequence 1 to 3 will be performed twice, then the sequence 5 to 15_8 will be performed and then both sequences will be repeated.

7.4.4 Execution

The program will execute the tests on the units specified above. When the sequence is complete the program will return to the 'TEST=' prompt. The program recognizes all of the break characters during execution.

7.5 Messages and Halts

Messages to the operator are printed on the console teletype. Messages are printed when either the status information or data input through the formatter differs from the expected value. The test operator can inhibit the output as required (with the EMSK=prompt).

7.5.1 Status Reporting

This program will print a status message when the received status differs from the expected status in bit positions enabled by the error mask word, EMSK. The message printed has the format:

```
HE   nnnn ssss cccc dddd mmmm
where:    nnnn is the state counter for the
          test in progress
          ssss is the status
          cccc is the command word
          dddd is the disk address
          mmmm is the memory address.
```

The bits in ssss include the 8 defined hardware status flags (values 1-200_8) and three software generated bits:

400_8	SKNE claimed an error when there were no error bits set in the status register or claimed no error when some status register error bit was set
1000_8	BUSY stayed true beyond the limits that should have caused a TMO error
2000_8	Timeout on a seek or restore operation, with BUSY false

7.5.2 Verify Error

When data disagrees (with or without hardware errors) a message is printed:

```
VE nnnn dddd wwwww=yyyy ggggg=zzzz
where:    nnnn is the test state
          dddd is the disk address
          wwwww is the address of the expected data,
          including the field digit
          yyyy is the expected data
          ggggg is the address of the data actually
          received
          zzzz is the data in error
```

Ordinarily, the right hand four digits of wwwww and gggg will be within the read and write buffers, but may take special values, e.g., 00055 and 00056 for the computer value of a checksum. The VE message will be repeated no more times on a given disk sector than the detail limit allows.

7.5.3 Program Halts

The program will halt during the operation of the interactive tests. The operator needs to depress RUN to continue the operation.

7.6 User Specified Test Mode

The test operator may respond to the 'PACK=' prompt with 'U' to enter the user specified test mode. The program will be put in a special mode of operation allowing the construction of test sequences tailored to a specific problem.

The operator can define up to eight test steps. Each step may be executed repeatedly until instructed otherwise or all the test steps may be repeated cylically. The operator builds the test steps by responding to the supplementary prompting messages, OP= and AD=.

7.6.1 OP=

<u>OP</u>=cmd, data type, expected status, mask, field

where:

cmd is the disk command including all modifier bits. This parameter must be entered.

data type specifies the pattern with which the buffer is filled before activating the disk. The choices for this parameter are:

0 -- 1 set all locations to 0
1 -- set all locations to 7777_8
2 -- set the buffer to alternate 5252_8 and 2525_8
3 -- set each location to an ascending sequence
4 -- set each location to a descending sequence

5 -- set each location to a pseudo-random number

6 -- set all locations to 6666_8 the worst case pattern

The default is 0.

expected status may be an octal value and defaults to 0.

Mask is the error mask and defaults to 7777_8.

Field specifies the memory field for the coupler to use.

This is useful primarily for reading, since the data pattern is always loaded into field 0, and writing from another field would not access the desired data pattern.

The parameter may range from 0 to 7 and defaults to 0.

7.6.2 AD=

AD=unit, sector, X1, X2

where:

unit selects the drive to be used, 0 to 5. This parameter must be entered.

sector is the block number, track and sector, used by the formatter. The range is 0 to 4623 ~~2317~~$_8$ and defaults to 0.

X1 and X2 are the exclusive-OR masks for the software CRC generation. The computer checksum bits will be inverted whenever there is a '1' in the corresponding bit of X1 or X2. X1 corresponds to the most significant 12 bits of the CRC and X2 corresponds to the least significant 4 bits.

The operator can respond to the OP and AD message up to eight times. The operator ends the definition phase by entering a RETURN to the OP prompt. The program then enters the execution mode.

The operator controls the execution of the tests by using the console teletype and the panel switch register. The panel switch register will be accessed by the program. The program will test bits 0 and 1 for the following conditions:

Bit 0 -- If 0, the program halts before executing the next test but after filling the data buffer.
If 1, the program executes the test step without halting.

Bit 1 -- If 0, the program loops on the current test.
If 1, the program advances cyclically to the next test step in the list.

The break characters, Control-N, Control-P, and Control-F will also be recognized.

7.7 Test Section

The diagnostic program contains 23_8 tests. The tests sequence defaults to test 1 through 20_8. Following is a list of the tests. A more complete description of each follows.

TEST NUMBER	NAME
0	Verify Read
1	Format Sectors
2	Verify Sector Addresses
3	Seek and Restore
4	Half Sector Write
5	Half Sector Read
6	CRC Generation
7	Memory Addressing
10	Worst Case Data
11 and 12	Random Data Tests
13	Memory Field
14	CRC Error Forcing
15	Interrupt Function
16	Status Interference
17	Timeout
~~20~~	~~Preamble Compare Error~~
2~~1~~0	Write Protect
2~~2~~1	Not Ready

7.7.1 Test 0 -- Verify Read

Function: To verify that the disk pack is readable and that the preamble, data and checksum are correct.

Procedure: The test will check-read the contents of the disk in sequence as defined in the 'PACK=' specifiers. The test expects a data error (bit 7 in status register) but no other errors. If any other errors are detected they are printed. Besides reporting the errors a check is made that the memory verify area was not changed. A verify error, if reported, means that data was transferred to memory.

7.7.2 Test 1 -- Format Sectors

Function: To format a new disk pack or prepare a disk for a series of diagnostic tests. There is no prerequisite for test 1.

Procedure: ~~The test will use the write buffer~~ ~~as follows:~~

~~0 preamble~~

~~1-256 all zeros~~

~~257,258 CRC value~~

~~The data in word 0 is incremented for sector.~~ The test will issue the ~~Write Preamble and CRC~~ Format Track (Diagnostic Write with 1/2 sector bit on) command to the formatter for each track in the PACK specification.

7.7.3 Test 2 -- Verify Sector Addresses

Function: To ensure unique, correct addressability of each unit, track and sector.

Procedure: This test writes the disk address to the first word of each sector. The test reads the preamble, data, and CRC (259 total words of information) into the read buffer, and verifies its accuracy. This test must be run before using test 3.

7.7.4 Test 3 -- Seek and Restore

Function: To check the head positioning logic.

Procedure: A pseudo-random sequence of track address are generated (the same sequence repeated for each PACK specified). For each track address, two seeks are issued. The first is to the given track from the current position. The second is to the given track from the given track with a restore preceeding the actual seek. With each seek the sector is read, Word 0 contains the block count that is used to verify the accuracy of the read.

7.7.5 Test 4 -- Half Sector Write

Function:	To test the half sector write logic.
Procedure:	A 256-word buffer is set with a 1's pattern. A sector is written in the 'half-sector' mode and checked to see that the first part of the data was written correctly, and the rest of the sector was padded with 0's as required. Then write a full sector of data and verify that the half sector mode bit is reset.

7.7.6 Test 5 -- Half Sector Read

Function:	To test the half sector read logic.
Procedure:	A full sector of 1's are written, read and compared. The upper half of the read buffer is cleared. A half sector read is then performed. The write buffer is then modified by clearing the upper half. The two buffers are then compared.

7.7.7 Test 6 -- CRC Generation

Function:	To validate the CRC generator in the controller.
Procedure:	Using each of the standard data patterns write a sector. After each write, read the sector with the Read Preamble and CRC command. The expected CRC is calculated by software and compared to the CRC read. Any verification error is printed.

7.7.8 Test 7 -- Memory Addressing

Function:	To test that correct memory addresses are generated during data transfer.
Procedure:	A sector of data is written with the ascending count pattern and then read back to every possible buffer area between the top of the program and location 7600_8.

7.7.9 Test 10 -- Worst Case Data

Function:	To check the recording surface integrity and data recovery.
Procedure:	The worst case data pattern is written in each PACK specification, read back and verified.

7.7.10 Tests 11 and 12 -- Random Data Tests

Function: To test the random addressing capability.

Procedure: Test 11 writes and reads pseudo-random data in each sector on every tack of the unit specified. Test 12 uses the data from test 11 to randomly select sectors to read back the data for verification. The random selection is repeated for 256 sectors.

7.7.11 Test 13 -- Memory Field

Function: To test the formatter's ability to address memory in different memory fields.

Procedure: The test verifies that data can be written and read into all eight addressable memory fields, and verifies that memory address incrementing from 7777_8 to 0 generates a corresponding field increment.

7.7.12 Test 14 -- CRC Error Forcing

Function: To verify the error detection for every bit position in the CRC logic.

Procedure: The test forces errors in all bit positions of the CRC by using the program generated CRC and the Write Preamble and CRC command. The program reads the sector to check that a CRC error bit is set in the status word. The program repeats the test for all standard data patterns.

7.7.13 Test 15 -- Interrupt Function

Function:	To test the coupler interrupt logic.
Procedure:	The test will operate from state 0 to state 7. Each state checks various interrupt conditions and reports an error if the correct condition has not occurred. The error message has the format:

```
     IE nnnn cccc
where: nnnn is the test state
       cccc is the disk command.
```

State 0:	Test SKNI when an operation is done but interrupts are inhibited.
State 1:	Test SKNI after SKNI to see that second time is always a skip.
State 2:	Test SKNI when an operation is done and interrupts are allowed.
State 3:	Test that SKNI cleared interrupt request although the CPU did not recognize the request.
State 4:	Test that interrupt occurs under the conditions that in State 2 caused SKNI to report an interrupt request.
State 5:	Test that SKNI works after the interrupt generated in State 4.

State 6: Test that a No-Op command with the interrupt-enable bit OFF, can clear a pending interrupt without the help of SKNI.

7.7.14 Test 16 -- Status Interference

Function: To test the entire mix of CPU data requests with the programmed status interruption.

Procedure: The test will verify that the reading of the status by the Read Status command during a write and then a read does not cause cross talk.

7.7.15 Test 17 -- Timeout

Function: To test the TMO status bit.

Procedure: The test attempts to read from a non-existant block thus forcing a TMO error. The TMO bit should be set.

~~7.7.16 Test 20 -- Preamble Compare Error~~

~~Function: To test the preamble compare error status bit.~~

~~Procedure: The program will do a Write Preamble and CRC for a sector. The preamble word will be in complemented form in the write buffer. The same sector is then read. A preamble compare error is expected. The preamble is then rewritten in its correct form.~~

7.7.16 Test 20-- Write Protect

Function:	To test the write protect logic.
Procedure:	The test will attempt to write on a protected pack. The diskette is inserted after the message 'TEST#22' is printed. Press RUN after inserting a protected pack.

7.7.17 Test 21 - Not Ready

Function:	To test the ready status logic.
Procedure:	The program halts to allow the operator to disable the primary drive, checks for not ready on an attempted read and then monitors the status until the drive is ready.